REMARKS/ARGUMENTS

Applicants thank the Examiner for taking the time to discuss the issues raised in the Office Action in a telephonic interview on November 6, 2003. This paper is in response to the Final Office Action dated August 14, 2003, the finality of which has been withdrawn, and to the interview.

Claims 1-9 and 11-20 are pending in the present application. Claims 1, 7-9, and 18-20 have been amended. Accordingly, claims 1-9 and 11-20 remain pending in the present invention.

Amended Claims

Applicants have amended independent claims 1 and 18-20 to recite that "a plurality of tree structures" are defined via the GUI, "wherein at least one tree structure represents an input data structure and at least one other tree structure represents an output data structure." Support for the amendment is found in claim 6 and throughout the Specification. Claims 7-9 were amended to provide proper claim dependency. No new matter has been presented by the amendments.

35 U.S.C. § 102 Rejections

The Examiner rejected claims 1, 4-6, 8-9, 13, 16, and 18-20 under 35 U.S.C. §102(e) as being anticipated by Lee et al. (U.S. Patent No. 6,535,883). In so doing, the Examiner stated:

> Referring to claims 1 and 18-20, Lee discloses a tool for graphically defining an expression with a graphic user interface, wherein the user input for defining tree structures comprising a hierarchical series of nodes, and one or more lists comprising a plurality of items, each item being associated with a respective node of an associated tree structure, wherein the tree structure represents an output data structure, as shown in Figure 10, the associated list items being the field names and means for formatting the definition of these field names (column 3, lines 4-19). Lee further discloses an expression generator component (reference

number 490, Figure 11), which is adapted to read a graphic definition of an expression provided by a user through said GUI component, the graphic being the node from the tree structure to which these expression apply (reference number 460, Figure 11), and means for analyzing the graphic definition, wherein an expression is generated based on the structure of each tree and any list items associated with the respective nodes of a tree, wherein the expression which best associates with the structure of the tree and the placing of the list items is chosen to best express the graphic definition (Figure 11 and column 3, lines 43-53).

Applicants respectfully disagree. The present invention is directed to a method and tool for graphically defining an expression, in particular, a query containing complex or non-native data types. In the prior art, graphical user interface tools allow a developer to formulate expressions for configuring either filters for incoming messages (in a messaging and queuing system) or database queries. Those tools, however, only allow users to define values for simple data types.

According to the preferred embodiment of the present invention, the tool includes a graphic user interface (GUI) that allows a user to define a plurality of tree structures comprising a hierarchical series of nodes. At least one of the tree structures represents an input data structure and at least one other tree structure represents an output data structure. The user also is allowed to define one or more lists that include a number of items, whereby each item on a list is associated with a node on a tree. Any item associated with the tree structure representing the input data structure is a filtering constraint and any item associated with the tree structure representing the output data structure defines a formatting definition.

Once the user has defined the plurality of tree structures, an expression generator analyzes the tree/list structure and generates an expression. The expression is used to configure modules in a relational message broker or to configure a database query.

The present invention, as recited in claims 1 and 18, provide:

1. A tool for graphically defining an expression, said tool comprising:
a graphic user interface (GUI) component comprising:

means, responsive to user input, for generating a graphic definition of the expression by defining a plurality of tree structures comprising a hierarchical series of nodes, and one or more lists comprising a plurality of items, each list item being associated with a respective node of an associated tree structure, wherein at least one of the tree structures represents an input data structure and at least one other tree structure represents an output data structure wherein any associated list item defines a formatting definition;

an expression generator component adapted to read the graphic definition of the expression provided by a user through said GUI component, expression generator component comprising:

means for analyzing said graphic definition and generating an expression based on the structure of each tree and any list items associated with respective nodes of a tree.

18. A method for graphically defining an expression in accordance with a graphic definition comprising the steps of:

(a) defining a plurality of tree structures comprising a hierarchical series of nodes, and one or more lists comprising a plurality of items responsive to user input, each list item being associated with a respective node of an associated tree structure, wherein at least one of the tree structures represents an input data structure and at least one other tree structure represents an output data structure wherein any associated list item defines a formatting definition;

(b) analyzing said definition; and

(c) generating an expression based on the structure of each tree and any list items associated with respective nodes of a tree.

Claims 19 and 20 are computer product and system claims, respectively, having similar scopes to that of claim 18.

In contrast to the present invention, Lee is directed to creating a set of validation rules for a group of fields in an electronic form. The form is graphically represented as a tree with nodes representing fields to be filled out in the form. Validation rule(s) for a field govern what type of information can be entered in that field. The validation rule(s) are represented as subnode(s) of the field.

Lee fails to teach or suggest generating a graphic definition of an expression by "defining a *plurality* of tree structures" "wherein at least one of the tree structures represents an input data structure and at least one other tree structure represents an output data structure wherein any associated list item defines a formatting definition," as recited in claims 1, 18-20. Moreover, Lee fails to teach or suggest "generating an expression based on the structure of each tree and any list items associated with respective nodes of a tree," as recited in claims 1, 18-20. In Lee, *a single tree structure* is generated to represent the electronic form. In other words, the graphic definition of Lee's electronic form is one, and only one, tree. Lee does not define "a plurality of tree structures" that together provide a graphic definition of an expression, as recited in claims 1, and 18-20.

Moreover even if Lee disclosed defining a plurality of trees, which Applicants respectfully submit it does not, Lee's tree structure fails to represent "an output data structure wherein any associated list item defines a formatting definition." In the present invention, the output data structure is a tree that defines what information should be outputted and in what format that information should be outputted. Thus, the nodes of the output tree structure define the type of output information and the list item(s) associated with each node defines the format. In Lee, the tree structure defines what data should be *inputted*, not outputted, and in what format that data should be *entered*, not outputted. Thus, Applicants respectfully submit that Lee actually teaches away from the present invention in that Lee's tree structure fails to teach or suggest "an output data structure wherein any associated list item defines a formatting definition," as recited in the independent claims.

In addition, Lee provides no teaching or suggestion of taking "the graphic definition of an expression provided by a user" and "generating the expression based on the structure of each tree and any list items associated," as recited in the independent claims. In Lee, the user generates a set of validation rules for a field by choosing expressions from a list displayed by a GUI and filling in the appropriate conditions. Once created, the validation rules become a part of the tree structure representing the electronic form. Lee's lone tree structure is the graphic definition of the electronic form. Lee fails to teach or suggest "generating the expression based on the structure of *each* tree," as recited in the independent claims. Accordingly, Applicants respectfully submit that claims 1, and 18-20 are allowable over Lee.

In the Office Action, the Examiner contends that the "output tree structure" of the present invention is taught by Lee's tree structure in Figure 10. The Examiner asserts that the associated list items are "the field names and means for formatting the definition of these field names (column 3, lines 4-19)." During the interview, the Examiner further explained that Figure 10 depicts an "output tree structure" because it is a tree structure that is *displayed* to the user via the GUI, and that each field name (associated with each node) is a "formatting definition" because it define the nodes. Applicants respectfully disagree.

According to Lee, Figure 10 depicts the tree structure 430 *after* the user has completed the "new rule dialog box" (Figure 9), which defines first and second validation rules for the field (node) selected. Figure 10 illustrates the first and second rule nodes 460 and 470, which define: the validation level (460), the error message and override mode (470) (col. 8, lines 36-56). Like all tree structures in Lee, Figure 10 represents the electronic form and defines what data should be *inputted* and in what format the data should be *inputted.* It does not represent "an output data structure." Moreover, while each node in Lee's tree structure is defined by a field name (col. 3, lines 4-19), the field name merely identifies the node. The field name does not *define in what format* it should be outputted. Indeed, nothing in Lee teaches or suggests that the user has any ability to define *formatting definitions* for the field names.

In addition, the Examiner contends that the expression generator component of the present invention is taught by Lee's "add expression menu 490" in Figure 11. According to Lee, the "add expression menu 490" pops up after the user selects the "add expression" option in the action menu 480 (Figure 11). The add expression menu 490 "includes commonly-used expressions which are implemented as expression templates." (Col. 9, lines 5-25). "The expression template includes the test or comparison operator or operands that must be added to complete the expression. Thus, [Lee] allows a user to create a validation rule by selecting a template and fill in the blanks." (Col. 3, lines 43-53). Nothing in Lee teaches or suggests that the "add expression" pop-up menu 490 *analyzes* the "graphic definition" of an expression and generates "an expression *based on the structure of each tree and any list items associated* with respective nodes of a tree," as recited in claims 1, 18, 19 and 20.

Accordingly, based on the arguments above, Applicants respectfully submit that Lee fails to teach or suggest the cooperation of elements recited in claims 1, 18, 19 and 20. Thus, claims 1, 18, 19 and 20 are allowable over the cited reference. Claims 4-6, 8, 9, 13 and 16 depend on claim 1, and the above arguments apply with full force. Therefore, claims 4-6, 8, 9, 13 and 16 are also allowable over the cited reference.

35 U.S.C. §103 Rejections

The Examiner rejected claims 2, 7, 11, 14 and 15 under 35 U.S.C. §103(a) as being unpatentable over Lee in view of Premerlani et al (U.S. Patent No. 5,555,367). Claim 3 was rejected under 35 U.S.C. §103(a) as being unpatentable over Lee and Premerlani and further in view of Gawlick et al (U.S. Patent No. 6,377,953). Claim 12 was rejected as being unpatentable over Lee. Claim 17 was rejected as being unpatentable over Lee and Premerlani and further in view of Moshfeghi (U.S. Patent No. 6,476,833).

Claims 2, 3, 7, 11, 12, 14, 15 and 17 depend on claim 1 and therefore, the arguments with regard to claim 1 apply with full force to claims 2, 3, 7, 11, 12, 14, 15 and 17. Thus, even if the secondary references disclose the features described by the Examiner, claims 2, 3, 7, 11, 12, 14, 15 and 17 are still allowable because Lee fails to teach or suggest the present invention as recited in claim 1. Accordingly, Applicants respectfully submit that claims 2, 3, 7, 11, 12, 14, 15 and 17 are allowable over the cited references.

Conclusion

In view of the foregoing, it is submitted that the claims 1-9 and 11-20 are allowable over the cited references and are in condition for allowance. Applicants respectfully request reconsideration of the rejections and objections to the claims, as now presented.

Applicants believe that this application is in condition for allowance. Should any unresolved issues remain, Examiner is invited to call Applicants' attorney at the telephone number indicated below.

Respectfully submitted,
SAWYER LAW GROUP LLP

[illegible]

November 14, 2003
Date

Joyce Tom
Attorneys for Applicant(s)
Reg. No. 48,681
(650) 493-4540